Fault Injection Modeling Attacks on 65nm Arbiter and RO Sum PUFs via Environmental Changes

Jeroen Delvaux and Ingrid Verbauwhede, *Fellow, IEEE*

***Abstract*—Physically Unclonable Functions (PUFs) are emerging as hardware security primitives. So-called strong PUFs provide a mechanism to authenticate chips which is inherently unique for every manufactured sample. To prevent cloning, modeling of the challenge-response pair (CRP) behavior should be infeasible. Machine learning (ML) algorithms are a well-known threat. Recently, repeatability imperfections of PUF responses have been identified as another threat. CMOS device noise renders a significant fraction of the CRPs unstable, hereby providing a side channel for modeling attacks. In previous work, 65nm arbiter PUFs have been modeled as such with accuracies exceeding 97%. However, more PUF evaluations were required than for state-of-the-art ML approaches. In this work, we accelerate repeatability attacks by increasing the fraction of unstable CRPs. Response evaluation faults are triggered via environmental changes hereby. The attack speed, which is proportional to the fraction of unstable CRPs, increases with a factor 2.4 for both arbiter and ring oscillator (RO) sum PUFs. Data originates from a 65nm silicon chip and hence not from simulations.**

***Index Terms*—arbiter PUF, ring oscillator PUF, fault injection, modeling, repeatability, supply voltage, temperature.**

I. Introduction

THERE is a clear trend towards small, distributed, mobile and wireless applications. They are typically integrated on chip. Cryptographic protection is indispensable as almost all applications process sensitive data, but is thwarted because of the trend above. Energy/power and chip area are scarce resources, so we are often limited to lightweight cryptography. Furthermore, because of the mobility, one can easily gain physical access to the chip. Hardware attacks, either invasive or noninvasive, are thus a significant threat.

Classical cryptography heavily relies on the ability to store secret information. Traditionally, binary keys are stored in on-chip Non-Volatile Memory (NVM). EEPROM and its successor Flash are the main technologies. However, this approach is vulnerable to hardware attacks [14]. The permanent nature of storage worsens the problem as no limits are posed on the time frame of the attacker. Circuits that detect hardware invasion offer additional protection. Unfortunately they suffer from practical limitations. They might be expensive, bulky, battery powered, vulnerable to bypassing and/or not appropriate for lightweight environments.

Physically Unclonable Functions (PUFs) have been proposed as a more secure and more efficient alternative. PUFs measure the unique variability of physical objects. They can be manufactured in a variety of technologies: optical, acoustical, magnetical, electrical and so on. PUFs which can be integrated on chip, especially in CMOS technology, are by far the most relevant for commercial applications. The manufacturing variability of nanoscale structures is then quantified.

For PUFs, the secret is stored in intrinsic physical features of a chip, resulting in some remarkable security advantages in comparison to on-chip NVM. First, PUFs are often assumed to be resistant against invasive attacks. One can argue that invasion damages the physical structure of the PUF. Second, keys are inherently unique for each manufactured sample of a chip and there is no need to explicitly program them. Third, the key is only generated and stored in on-chip volatile memory when key-dependent operations have to be performed, as such posing limits on the attackers time frame.

A. Weak and Strong PUFs

PUFs are functions and produce a response when queried with a challenge. Responses and challenges are both binary vectors. PUFs are often subdivided in two classes, depending on the number of challenge-response pairs (CRPs) [13]. Weak PUFs have few CRPs and are typically utilized for on-the-fly secret key generation. PUF response bits are not directly usable as a secret key because they are noisy and possibly correlated. A so-called fuzzy extractor ensures a reproducible and uniformly distributed key by applying an error-correcting code and a cryptographic hash function respectively [1].

Strong PUFs have many CRPs, in the ideal case exponentially increasing with the required chip area, and offer more applications. It should be infeasible to capture all their CRPs in a reasonable time span. Although secret key generation is possible as well, the most prominent strong PUF application is CRP-based authentication of a chip. In an enrollment phase, the verifier collects arbitrary CRPs from the chip and stores them secretly. In the verification phase, the verifier picks a challenge and requests the PUF response again. The returned response should match the one in the database. To avoid the error-correction overhead, a few erroneous bits are typically tolerated hereby. To prevent replay attacks, the verifier should discard a CRP once it has been employed for authentication.

B. Security

We assume an attacker to have physical access to the PUF chip. Remember that PUFs are claimed to resist hardware attacks better than on-chip NVM. Restriction to an eavesdropping attacker would undermine the need for PUFs. As nanoscale manufacturing variations are uncontrollable, it is

Jeroen Delvaux and Ingrid Verbauwhede, department of Electrical Engineering, KU Leuven, Belgium, and iMinds. Address: ESAT/COSIC Kasteelpark Arenberg 10, B-3001 Leuven-Heverlee, Belgium. E-mail: firstname.lastname@esat.kuleuven.be.

infeasible to produce a clone of a PUF. A functional clone, copying the CRP behavior, is the main threat however. The security considerations differ per application.

For secret key applications, it is imperative to keep the responses on chip, just as the secret keys to which they are post-processed. Hardware attacks (invasive, through side channels and via fault injection) should be taken into account. As mentioned before, PUFs are often assumed to be resistant against the first category. Experimental evidence is generally lacking however, except for the coating PUF [17]. For SRAM PUFs, an invasive attack has even been demonstrated [3]. Electromagnetic radiation is an exploitable side channel for ring oscillator (RO) PUFs [10]. In addition to PUF circuits, fuzzy extractors might leak side channel information as well [11].

For strong PUFs in a CRP-based authentation application, CRPs can be obtained by anyone with physical access to the chip. The security arises from the CRP behavior unpredictability. It should be infeasable to construct a clone via a mathematical model. Modeling through Machine Learning (ML) algorithms, given a training set of CRPs, is a major threat. The arbiter PUF, which quantifies the variability of gate delays, can be modeled as such [9]. Variants of the arbiter PUF which introduce additional non-linearity (XOR, feed-forward, ...) provide more resistance but can still be modeled [12]. Hardware attacks on strong PUFs should be considered too, as they can facilitate modeling. We introduced the repeatability side channel attack and demonstrated its feasibility on 65nm arbiter PUFs [5].

So-called controlled PUFs enhance the security of CRP-based authentication via additional hardware [2]. Hereby, the response bits of the strong PUF are post-processed using a fuzzy extractor. Cryptographic hash functions are non-invertible and modeling vulnerabilities of the CRP behavior are hence hidden. Also, one could preprocess the challenges with a cryptographic hash function to counteract chosen-challenge attacks. However, the use of controlled PUFs poses two major problems. First, the increased hardware footprint undermines the potential for resource constrained applications. Second, the additional building blocks are not necessarily resistant to hardware attacks.

C. Variability and Noise

The distinction between variability and noise is essential for a good understanding of this paper. Both cause deviations with respect to the nominal behavior. Measurements of (structural) variability, originating from manufacturing processes, are reproducible. One can state that they are defined by spatial distributions (and orientations) of individual molecules of the solid materials. Noise however is a non-reproducible temporal phenomenon. Generally speaking, in electronic circuits, both variability and noise are undesired. PUF circuits measure variability, but are bothered by noise as well, as it reduces the repeatability.

Both variability and noise are technology dependent. They remain major design and manufacturing challenges, especially while downscaling dimensions according to Moore's law. Random Dopant Fluctuation and Line-Edge/Width Roughness are important sources of variability for CMOS devices [8]. White thermal noise and $1/f$ noise affect the CMOS channel current [7]. Interconnect is affected by Line-Edge/Width Roughness and white thermal noise too.

D. Environment

Environmental conditions like temperature and supply voltage affect PUF behavior. In the ideal case, PUFs therefore operate in a constant nominal environment, which is to be specified. However, one can not force an attacker to obey this specification. In this paper, we intentionally apply environmental changes for PUF modeling purposes.

E. Our Contribution

Recently, we identified repeatability imperfections of PUF responses as a side channel for modeling strong PUFs [5]. The presence of CMOS device noise, rendering a significant fraction of the CRPs unstable, is exploited. As a proof of concept, 65nm arbiter PUFs have been modeled, with accuracies exceeding 97%. However, more PUF evaluations were required than for state-of-the-art ML approaches.

In this paper, we accelerate repeatability attacks by increasing the fraction of unstable CRPs. We trigger response evaluation faults via environmental changes. A significant performance gain is obtained for arbiter and RO sum PUFs, both manufactured in 65nm CMOS technology. Data originates from a silicon chip and hence not from simulations. We also describe the attack capabilities for RO PUFs, employed in a CRP-based authentication application, although the application use cases are rather limited.

This paper is organized as follows. In section II, III and IV, we introduce arbiter, RO and RO sum PUFs respectively. The high-level architecture, implementation aspects and security risks are described for each. The repeatability model of PUF responses, capturing the impact of CMOS device noise, is resumed in section V. Attacks exploiting the former side channel are described in section VI for all three PUFs. Fault injection, accelerating modeling via environmental changes, is introduced in section VII. Side channel and fault injection results are compared in section VIII, for 65nm arbiter and RO sum PUFs. Suggestions for further work are given in section IX. Section X concludes the work.

II. Arbiter PUF

A. Architecture

Arbiter PUFs [9] measure structural variability via the propagation delays of logic gates, like ring oscillator [15] and glitch PUFs [16]. The high-level functionality is represented by figure 1. A rising edge propagates through two paths with identically designed delays. Because of nanoscale manufacturing variations however, there is a delay difference Δt_V between both paths. An arbiter decides which path 'wins' the race ($\Delta t_V \lessgtr 0$) and generates a response bit r.

The two paths are constructed from a series of k switching elements. Challenge bits c_i determine for each stage whether path segments are crossed or uncrossed. Each (binary) state

of each stage has a unique contribution to Δt_V. So challenge vector $\vec{c}$ determines the arbiter time difference Δt_V and hence the response bit r. The number of CRPs equals 2^k.

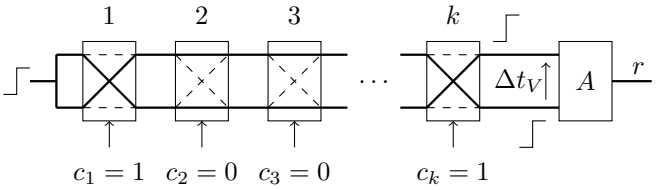

Fig. 1. Arbiter PUF.

CRP-based authentication with strong PUFs requires responses to have multiple bits. A single arbiter PUF produces only a single response bit however. Two solutions, or a mixture of both, are possible. First, one can implement multiple arbiter circuits on the same chip, all having the same challenge as input. Second, one can query a single PUF circuit with multiple challenges and concatenate the responses.

B. Implementation

Our 64-stage arbiter PUFs are manufactured in TSMC's 65nm Low Power CMOS technology [6]. Switching elements are implemented via two 2-to-1 inverting multiplexers, as shown in figure 2. As k is even, we still have two rising edges as arbiter input. Each multiplexer is implemented by two transmission gates and a static complementary CMOS inverter to restore signal level.

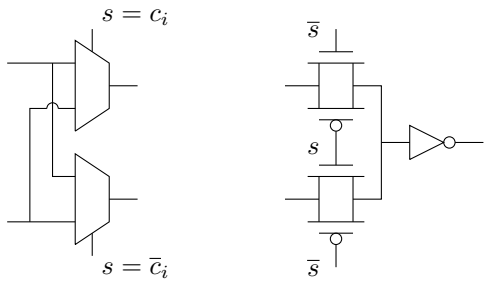

Fig. 2. Switching element circuit.

A variety of circuits can serve as an arbiter. Our chip employs a NAND latch, as shown in figure 3. Two cross-coupled NAND gates, implemented in static complementary CMOS logic, determine and store the response bit r. Initially inputs i_1 and i_2 are both zero so that memory nodes r and $\overline{r}$ are both charged. A rising edge will discharge one memory node and simultaneously lock the other.

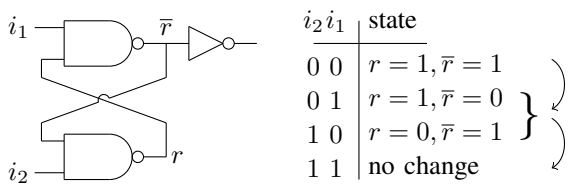

Fig. 3. Arbiter circuit: NAND latch.

It is important to match the delay of both NAND gates. Otherwise, bias is introduced and the response bit generation degrades to $\Delta t_V \lessgtr \Delta t_B$, with Δt_B a nonzero constant. The probability of r to be 1 (or 0) would not be 50% anymore, assuming a symmetrical Probability Density Function (PDF) of Δt_V with mean zero. Our arbiters are biased more than usual because response readout logic is connected to node $\overline{r}$ only, so that is has a higher capacitive load than node r. A dummy inverter connected to node r would improve the bias characteristics. A slight bias will always be present however as manufacturing variations prevent the NAND gates from being perfectly identical. So any modeling attack claiming to be general, should incorporate bias imperfections somehow, which we are able to demonstrate well for our attacks due to the forgotten replica.

C. Vulnerability to Modeling Attacks

Arbiter PUFs show additive linear behavior which makes them vulnerable to modeling attacks. A single stage can be described by two parameters, one for each challenge bit state, as illustrated in figure 4. The delay difference at the input of stage i flips in sign for the crossed configuration and is incremented with δt_i^1 or δt_i^0 for crossed and uncrossed configurations respectively.

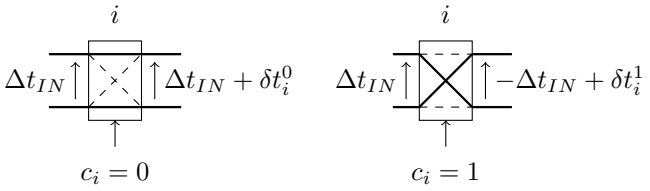

Fig. 4. Arbiter PUF: single stage modeling.

The impact of a δt on Δt_V is incremental or decremental for an even and odd number of subsequent crossed stages respectively. By lumping together the δt's of neighboring stages, one can model the whole arbiter PUF with only $k+1$ independent parameters (and not $2k$). A formal expression for Δt_V is as follows [12]:

$$\Delta t_V = \vec{\gamma}\vec{\tau} = \begin{pmatrix}\gamma_1 & \gamma_2 & \dots & \gamma_k & 1\end{pmatrix}\begin{pmatrix}\tau_1 & \tau_2 & \dots & \tau_{k+1}\end{pmatrix}^T$$

$$\text{with } \vec{\tau} = \frac{1}{2}\begin{pmatrix} \delta t_1^0 + \delta t_1^1 & + & \delta t_1^0 - \delta t_1^1 \\ & \vdots & \\ \delta t_{k-1}^0 + \delta t_{k-1}^1 & + & \delta t_k^0 - \delta t_k^1 \\ \delta t_k^0 + \delta t_k^1 & & \end{pmatrix} \text{ and}$$

$$\vec{\gamma} = \begin{pmatrix} (1-2c_1)(1-2c_2)\dots(1-2c_{k-1})(1-2c_k) \\ (1-2c_2)\dots(1-2c_{k-1})(1-2c_k) \\ \vdots \\ (1-2c_{k-1})(1-2c_k) \\ (1-2c_k) \\ 1 \end{pmatrix}^T .$$

Vector $\vec{\gamma} \in \{\pm 1\}^{1\times(k+1)}$ is a transformation of challenge vector $\vec{c} \in \{0,1\}^{1\times k}$. Vector $\vec{\tau} \in \mathbb{R}^{(k+1)\times 1}$ contains the lumped stage delays. Arbiter bias can be incorporated too, so that the response bit is still the outcome of $\Delta t_V \lessgtr 0$:

$$\tau_{k+1} = \delta t_k^0 + \delta t_k^1 - \Delta t_B.$$

D. Machine Learning

High modeling accuracies can be obtained through ML techniques like support vector machines and artificial neural networks. Given a limited set of training CRPs, algorithms automatically learn the input-output behavior by trying to generalize the underlying interactions. The more linear a system, the easier to learn its behavior. By using $\vec{\gamma}$ instead of $\vec{c}$ as ML input, a major source of non-linearity is eliminated. The non-linear threshold operation $\Delta t_V \lessgtr 0$ remains however.

In the paper proposing arbiter PUFs as a security primitive, ML was already identified as a threat [9]. They reported a modeling accuracy of 97% for their 64-stage $0.18\mu m$ CMOS implementation. A more recent $65nm$ implementation, having 64-bit challenges too, has been modeled equally accurate [4]. Our repeatability attacks are performed on the same $65nm$ chip. We circumvent the $\Delta t_V \lessgtr 0$ binarization by exploiting response repeatability as a side channel. A full linear system is obtained, which is straightforward to model.

III. Ring Oscillator PUF

A. Architecture

RO PUFs [15] quantify the manufacturing variability of k identically laid-out oscillators. The high-level functionality is represented by figure 5. A pairwise frequency comparison ($\delta f \lessgtr 0$) generates a single response bit r. Each challenge is hence defined by a pair of RO indices. Frequencies are typically measured by counting rising or falling edges on a wire connecting two subsequent inverters.

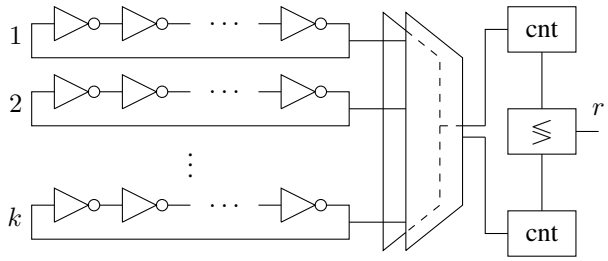

Fig. 5. Ring oscillator PUF.

B. Implementation

Our RO PUF is implemented on the same 65nm CMOS chip [6] as the arbiter PUFs. Each oscillator consists of 40 inverters and one NAND-gate, the latter being able to enable/disable the oscillation. For improved testability, the oscillation time can be configured and counter values are directly accessible (frequency comparisons are performed in software). This should not be the case however for market products claiming to be secure.

The oscillation time is regulated by a separate RO, incrementing its dedicated counter. Once a user-defined value is reached for this counter, all ROs are disabled and stop oscillating. The more oscillations one allows, the more stable the counter values, the more repeatable the PUF responses. We performed an experiment to determine an appropriate oscillation time. Figure 6 plots the noise-induced spread on RO counter values as a function of the oscillation time. The counter value spread for a single RO is quantified by the normalized standard deviation σ_{cnt}/μ_{cnt}, computed for 20 measurements, with μ_{cnt} the mean value. To improve figure quality, the spread of 8 ROs is averaged. For all further results, we configured a value of about 1000 oscillations, as indicated by the arrow. Responses are fairly repeatable then, without increasing energy consumption and PUF evaluation time unnecessarily.

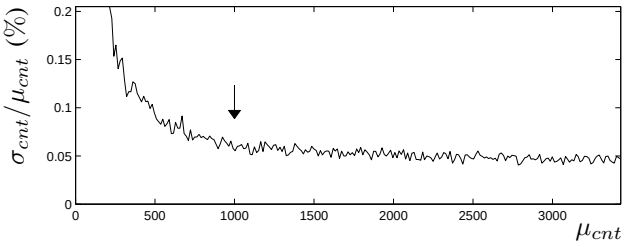

Fig. 6. RO PUF: trade-off between repeatability and oscillation time.

C. Vulnerability to Modeling Attacks

For CRP-based authentication, RO PUFs offer little security. Their number of challenges, which equals $k(k-1)/2$, increases quadratically and not exponentially with the PUF circuit area. As a consequence, it is always feasible to collect all CRPs in a reasonable time span. By brute force, one is thus able to construct a complete CRP table serving as a PUF clone. Even worse: not all CRPs have to be collected as they are interdependent. The total PUF entropy is only $log_2(k!)$ bit as there are $k!$ ways to sort the frequency values. Sorting algorithms with complexity $O(k\ log_2(k))$ accelerate the attack [12].

Therefore, only secret key generation is recommended as PUF application. Hereby, a fixed sequence of challenges is applied via an internal challenge generator and response bits are kept secret. For the sake of completeness, we will discuss repeatability attack for CRP-based authentication, although rather briefly.

The multiplexer-counter-comparator architecture might greatly vary in practice. The corresponding degree of parallelism affects the security against side channel attacks: measuring frequencies simultaneously offers more protection. Actually, there is a whole spectrum in between the following two extremes: an individual counter per RO and a single counter accessing all ROs via a giant multiplexer. Consider for instance the semi-invasive modeling attack proposed in [6], where one measures oscillator frequencies via the electromagnetic side channel. If one can not assign these frequencies to individual ROs, the attack is obstructed as such.

IV. RO SUM PUF

A. Architecture

RO sum PUFs [18] provide a much larger challenge space than their originals. The high-level functionality is represented by figure 7. ROs are subdivided into k pairs, with each pair having a certain frequency difference δf (digitized via counters). All δf's are summed, with challenge bits determining the individual signs as expressed below. A thresholding step binarizes Δf_V to a response bit. The number of CRPs equals 2^k.

$$\Delta f_V = \sum_{i=1}^{k} (1 - 2c_i)\delta f_i.$$

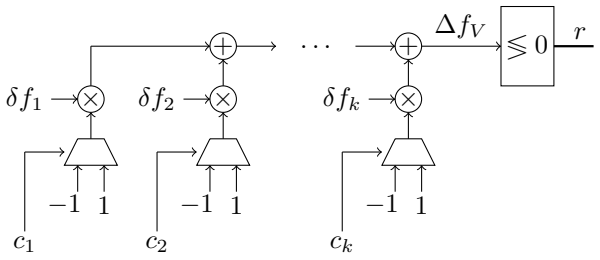

Fig. 7. RO sum PUF.

B. Implementation

The same array of 65nm on-chip ROs, selectively incrementing a counter via a multiplexer, is employed as analog PUF core. We choose $k = 64$. Similar as for RO PUFs, CRP evaluation is performed partly in software. The oscillation time parameter is chosen the same.

C. Vulnerability to Modeling Attacks

More obvious as for arbiter PUFs, there is additive linear behavior. RO sum PUFs are completely described by $k + 1$ independent parameters, including one dedicated parameter to model bias effects. Using the same notation as for arbiter PUFs, we obtain:

$$\Delta f_V = \vec{\gamma}\vec{\tau} \text{ with } \vec{\tau} = \begin{pmatrix} \delta f_1 & \delta f_2 & \dots & \delta f_k & -\Delta f_B \end{pmatrix}^T$$
$$\text{and } \vec{\gamma} = \begin{pmatrix} (1-2c_1) & (1-2c_2) & \dots & (1-2c_k) & 1 \end{pmatrix}.$$

Again, the thresholding operation is the only non-linearity in the system. The authors of [18] already indicate ML as a threat, although an attack has not been demonstrated. Later work on this PUF therefore concentrates on secret key applications instead of CRP-based authentication. We still consider this PUF to be an interesting test specimen for demonstrating the capabilities of our repeatability attacks.

V. PUF REPEATABILITY MODEL

Repeatability refers to the short-term reliability of a PUF as affected by CMOS (and interconnect) noise sources. Long-term device aging effects are not included. With $R \in [0, 1]$, we denote the fraction of the responses which evaluates to '1' for a certain CRP. The further from $R = \frac{1}{2}$, the more repeatable.

A. Model Description

A generalized PUF architecture is shown in figure 8. There is always an analog circuit harvesting variability, which is unfortunately affected by CMOS device noise and the environment too. For RO and especially RO sum PUFs, there is additional post-processing logic, which is digital and hence fully deterministic. A signal Δx, having mean zero in the ideal case, is binarized to a response bit using threshold zero, also in the ideal case. Bias might be introduced before or during binarization, but in both cases the response bit evaluates as $\Delta x \lessgtr \Delta x_B$. Without loss of generality, we develop our model for the latter case.

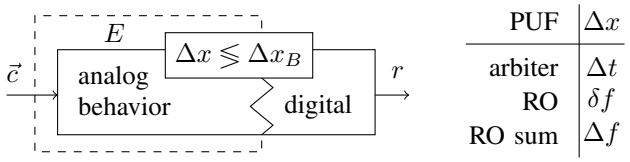

Fig. 8. Generalized PUF architecture.

The PUF repeatability model is represented by figures 9 and 10, the former for the ideal case $\Delta x_B = 0$ only. We assume variability and noise components to be independent so that $\Delta x = \Delta x_V + \Delta x_N$. As both components are the result of very complex processes, we expect them to be normally distributed according to the central limit theorem. Variance component Δx_V is normally distributed with respect to the set of all challenges, as shown in subfigures a. We assume a zero mean and denote its standard deviation as σ_V. Arrows indicate Δx_V for a particular challenge $\vec{c}_i$.

We collect the device noise relevant for response bit generation in one equivalent noise source. Noise component Δx_N is normally distributed with respect to the infinite set of all PUF evaluations. We assume a zero mean and denote its standard deviation as σ_N. Subfigures b show the PDF of $\Delta x(\vec{c}_i)$, with variance and noise components determining its mean and its spread respectively. The fraction which corresponds to a response $r = 0$, is marked black. Repeatability R, as shown in subfigures c, is computed by integrating the former PDF:

$$R(\Delta x_V) = \frac{1}{2} erfc\left(\frac{\Delta x_B - \Delta x_V}{\sqrt{2}\sigma_N}\right)$$
$$\text{with } erfc(x) = \frac{2}{\sqrt{\pi}} \int_x^\infty e^{-z^2}\, \mathrm{d}z.$$

The key insight is that repeatability measurements provide direct access to internal analog information, as expressed below. Knowledge of neither σ_V nor σ_N is required for PUF modeling purposes. Acquired analog information is all relative, which is not a problem as in the end we only need the sign of $\Delta x_V - \Delta x_B$ to predict response bits.

$$\Delta x_V(R) = \Delta x_B - \sqrt{2}\sigma_N erfc^{-1}(2R).$$

B. Repeatability Measurements

Fraction R can be estimated by evaluating the same challenge multiple times (parameter M). A measurement error ϵ_R is bound to be present. As a consequence, an error $\epsilon_{\Delta x_V}$ on

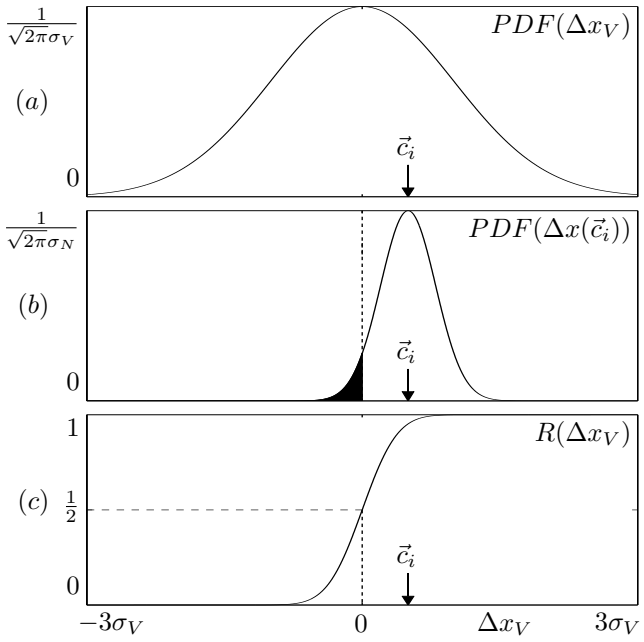

Fig. 9. PUF repeatability model, bias excluded. (a) Normal distribution of variance component Δx_V. (b) Normal distribution of noise component Δx_N, superimposed onto a particular challenge $\vec{c}_i$. (c) Repeatability R.

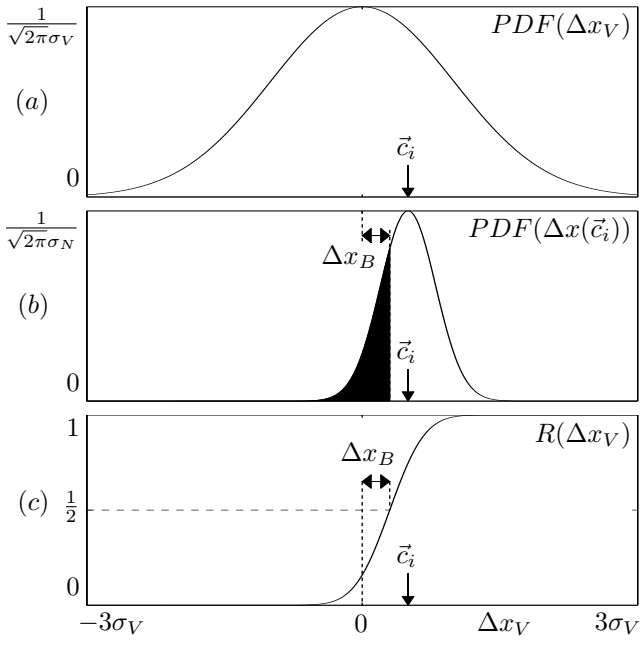

Fig. 10. PUF repeatability model, bias included. (a) Normal distribution of variance component Δx_V. (b) Normal distribution of noise component Δx_N, superimposed onto a particular challenge $\vec{c}_i$. (c) Repeatability R.

the estimated value of Δx_V is present too. For small ϵ's, the derivative $\frac{d\Delta x_V}{dR}$ serves as a scaling factor:

$$\epsilon_{\Delta x_V} = \sqrt{2\pi}\sigma_N exp\left(\left(erfc^{-1}(2R)\right)^2\right)\epsilon_R.$$

We distinguish two error phenomena making ϵ_R nonzero. First, there is the discretization $R \in \{0, \frac{1}{M}, \ldots, \frac{M-1}{M}, 1\}$. The larger M, the less significant this type of errors. Second, there are stochastic errors. We consider a single PUF evaluation as a Bernoulli trial; multiple evaluations then describe a binomial distribution. For simplicity, we could define stochastic error ϵ_R as the standard deviation of the random variable R:

$$\epsilon_R = \sqrt{\frac{R(1-R)}{M}}.$$

Stochastic measurement error ϵ_R has a maximum at $R = \frac{1}{2}$ and decreases monotonically towards $R = 0$ and $R = 1$. Scaling factor $\frac{d\Delta x_V}{dR}$ has an opposing effect and is the most dominant. It has a minimum at $R = \frac{1}{2}$. Towards $R = 0$ and $R = 1$, it increases monotonically and approaches ∞ asymptotically. We prefer measurements around $R = \frac{1}{2}$, but consider the whole $10 - 90\%$ region as reasonable.

C. Model Validation

We provide a visual validation for our repeatability model. Analytical PDF expressions are matched with a normalized histogram of experimental data. For RO PUFs, we obtain variability and noise characteristics directly from the counter values. We evaluated 2048 ROs with $M = 15$. A histogram of averaged counter values μ_{cnt} is shown in figure 11a. We overlay a normal distribution with the same mean and standard deviation. A histogram of counter value deviations, with respect to the averaged values, is provided in figure 11b. The former two plots validate the model for both RO and RO sum PUFs.

For the arbiter PUF, we validate our model via the PDF of fraction R. An analytical expression is given below. We measured the reliability of one PUF circuit for 65000 random challenges with $M = 2000$. A nonlinear curve fitter iterating over two variables, σ_N/σ_V and $\Delta t_B/\sigma_V$, provides the match. Figure 11c shows an overlay of both PDFs. Only data in the $10 - 90\%$ region has been used for better visibility. Also note the minor bias towards $r = 0$, as discussed before in section II-B.

$$PDF(R) = PDF(\Delta t_V(R))\left|\frac{d\Delta t_V}{dR}\right| = \frac{\sigma_N}{\sigma_V} exp\left(\right.$$
$$\left.(erfc^{-1}(2R))^2 - \frac{1}{2}\left(\frac{\Delta t_B - \sqrt{2}\sigma_N erfc^{-1}(2R)}{\sigma_V}\right)^2\right).$$

VI. Modeling Attacks Exploiting CMOS Noise

Arbiter and RO sum PUFs show a similar form of additive linear behavior and are modeled likewise. We discuss them first. Remark that each of their response bits contains variability contributions from the whole PUF circuit. Next, RO PUFs are briefly discussed. Each of their response bits is affected by only a very small fraction of the whole PUF circuit. This turns out to be fundamentally different from an attacker's point of view.

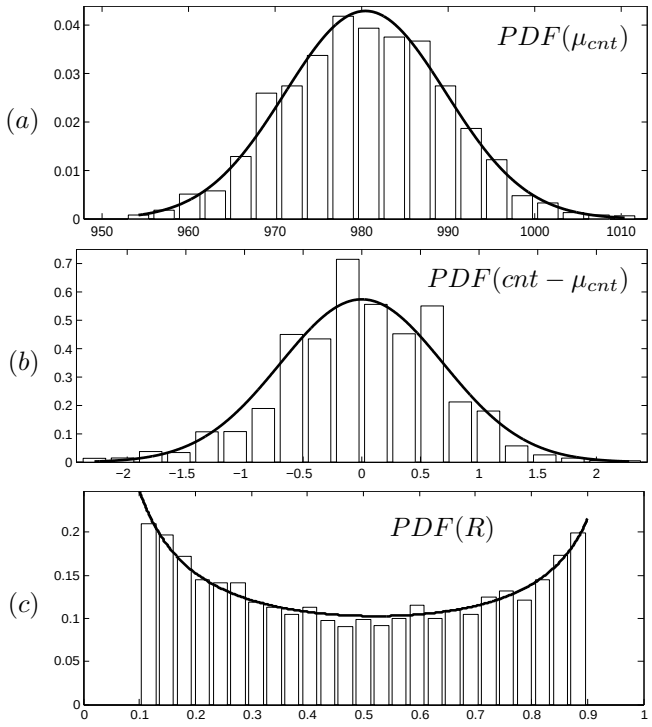

Fig. 11. Repeatability model validation. (a) RO (sum) PUF: variance component. (b) RO (sum) PUF: noise component. (c) arbiter PUF.

A. Modeling Arbiter and RO Sum PUFs

Figure 10c shows that $R(\Delta x_V)$ is fairly linear for $10\% \leq R \leq 90\%$: we call this the linear region. As Δx_V is a linear combination of the model parameters (τ_1 to τ_{k+1} for both arbiter and RO sum PUFs), so is R in the linear region (approximately). Via the repeatability side channel, the PUF degrades to a full linear system, which is straightforward to model. Consider a set of N training CRPs, all in the linear region and evaluated M times each: $\{\vec{c}_i, R_i\}$. For $N \geq k+1$, we can simply solve the (overdetermined) system of linear equations shown below in a Least Mean Square manner. Numerically stable algorithms are described in literature.

$$\mathbf{\Gamma}\vec{\tau} = \begin{pmatrix} R_1 \\ R_2 \\ \vdots \\ R_N \end{pmatrix} \quad \text{with } \mathbf{\Gamma}^{N\times(k+1)} = \begin{pmatrix} \vec{\gamma}_1 \\ \vec{\gamma}_2 \\ \vdots \\ \vec{\gamma}_N \end{pmatrix}.$$

As discussed earlier, bias is included in element τ_{k+1}. To predict the PUF response for a challenge $\vec{c}$, one should check whether $R = \vec{\gamma}\vec{\tau} \lessgtr \frac{1}{2}$. The predicted value of R can also be utilized to estimate the prediction certainty: the further from $\frac{1}{2}$, the better. This feature is not intrinsically available for ML techniques like artificial neural networks.

One could improve the linearity by applying the transformation below. For response prediction, one should check whether $\vec{\gamma}\vec{\tau} \lessgtr erfc^{-1}(1) = 0$. We do not apply this transformation as we observe only very minor improvements for the modeling accuracy.

$$\mathbf{\Gamma}\vec{\tau} = \begin{pmatrix} -erfc^{-1}(2R_1) \\ -erfc^{-1}(2R_2) \\ \vdots \\ -erfc^{-1}(2R_N) \end{pmatrix}.$$

B. Accuracy Analysis

We now analyze the modeling accuracy of the former attack in a theoretical manner. Errors in the repeatability measurements ($\vec{\epsilon}_R$) cause an error $\vec{\epsilon}_\tau$ on the PUF model vector obtained by the attacker:

$$\mathbf{\Gamma}(\vec{\tau} + \vec{\epsilon}_\tau) = (\vec{R} + \vec{\epsilon}_R) \quad \text{with } \mathbf{\Gamma}\vec{\tau} = \vec{R}.$$

To quantify the extent to which repeatability errors propagate, one could compute the condition number κ_C of $\mathbf{\Gamma}$. In our case, a modified definition is much more appropriate. Model error $\vec{\epsilon}_\tau = \mathbf{\Gamma}^+\vec{\epsilon}_R$ is determined by the (pseudo)inverse of challenge matrix $\mathbf{\Gamma}$. Row i of $\mathbf{\Gamma}^+$ determines for τ_i to which extent repeatability errors accumulate/cancel out. To quantify the overall condition, we sum the elements in each row of $\mathbf{\Gamma}^+$ and subsequently average their absolute values:

$$\kappa_C = \frac{1}{k+1}\sum_{i=1}^{k+1}\left|\sum_{j=1}^{N}\mathbf{\Gamma}^+_{i,j}\right|.$$

To study the impact of N, we performed a simulation as shown in figure 12. For different values of $N \geq k+1$, we generated 1000 random challenge matrices and averaged their condition numbers. Condition number κ_C decreases monotonically with increasing N: rapidly in the beginning and increasingly slower afterwards. As demonstrated later in section VIII, very small values of M become feasible as measurement errors on R cancel out efficiently.

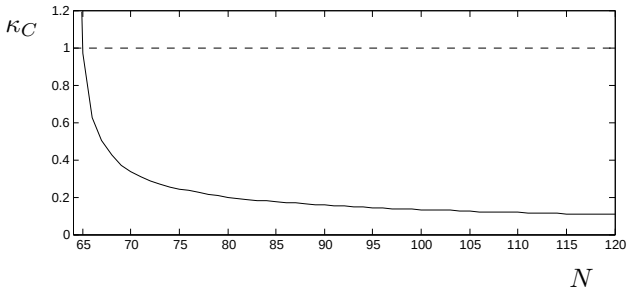

Fig. 12. Accuracy analysis of repeatability attacks on arbiter and RO sum PUFs: condition number simulation.

C. Query Algorithms

A major speed bottleneck is that most CRPs are very repeatable and hence not suitable for modeling purposes. One might try to increase the fraction of usable CRPs by an adaptive query algorithm. Hereby, the challenges are not chosen at random anymore. The ability to perform small steps of Δx_V turns out to be useful: a CRP originally in the linear

region might remain there after making a small step. Consider the following arbitrary challenge as a reference:

$$\begin{cases} \vec{c}_{REF} = (c_1 & c_2 & \dots & c_k) \\ \vec{\gamma}_{REF} = (\gamma_1 & \gamma_2 & \dots & \gamma_k & 1). \end{cases}$$

One can choose challenges $\vec{c}_i$ so that $\vec{\gamma}_{REF}$ and $\vec{\gamma}_i$ only differ in element i, with $i \in [1\ k]$:

$$\begin{cases} \vec{c}_1 = (\overline{c}_1 & c_2 & \dots & c_k) & \text{(arbiter PUF)} \\ \vec{c}_1 = (\overline{c}_1 & c_2 & \dots & c_k) & \text{(RO sum PUF)} \\ \vec{\gamma}_1 = (-\gamma_1 & \gamma_2 & \dots & \gamma_k & 1) \end{cases}$$

$$\begin{cases} \vec{c}_2 = (\overline{c}_1 & \overline{c}_2 & \dots & c_k) & \text{(arbiter PUF)} \\ \vec{c}_2 = (c_1 & \overline{c}_2 & \dots & c_k) & \text{(RO sum PUF)} \\ \vec{\gamma}_2 = (\gamma_1 & -\gamma_2 & \dots & \gamma_k & 1) \end{cases}$$

$$\vdots$$

$$\begin{cases} \vec{c}_k = (c_1 & \dots & \overline{c}_{k-1} & \overline{c}_k) & \text{(arbiter PUF)} \\ \vec{c}_k = (c_1 & \dots & c_{k-1} & \overline{c}_k) & \text{(RO sum PUF)} \\ \vec{\gamma}_k = (\gamma_1 & \dots & \gamma_{k-1} & -\gamma_k & 1). \end{cases}$$

Only τ_i then contributes to the difference in Δx_V. For optimal performance, one should learn while querying. Actually the principle above can also be used for an attack scheme which estimates the elements of model vector $\vec{\tau}$ one by one [5]. However, its performance and usability are in all aspects inferior to the method presented.

We do not implement a query algorithm because it has two major drawbacks. First, one should take into account the method by which multiple response bits are generated, as discussed in section II-A. Modeling a single PUF is more advantageous than modeling many parallel PUFs. That's because query algorithms can adapt their behavior to maximally benefit only a single PUF. Second, the rows of $\mathbf{\Gamma}$ become strongly correlated, which increases its condition number.

D. Modeling RO PUFs

Consider a RO PUF employed in a CRP-based authentication application. A single CRP evaluation does reveal the sign of a certain δf, with CMOS device noise causing an occasional error hereby. State-of-the-art modeling attacks require $O(k\ log_2(k))$ of these CRP evaluations. We claim that the complexity can be reduced to $O(k)$ via repeatability measurements. We explain the main idea, but we do not provide refinements or experimental results because of two reasons. First, we do not expect a substantial gain in terms of absolute numbers, if any, as k is limited. Second, as mentioned before, even a simple brute-force tabulation of all CRPs would be feasible in practice.

Repeated CRP evaluations do provide a (relative) estimate of a certain δf, instead of just the sign. The higher M, the better one can quantify the overlap of two counter PDFs and hence their δf. If there is no significant overlap, only a lower/upper bound can be provided. Consider a limited number of reference ROs, with all δf's known, as in figure 13a. We assume them to cover the whole frequency range, as suggested in figure 13b. The frequency of every RO can be determined with respect to this set, making the complexity of the attack $O(k)$. Because M should be rather high and because the number of CRPs is limited, both compared to the arbiter and RO sum PUF attack for instance, we do not expect to obtain a performance advantage.

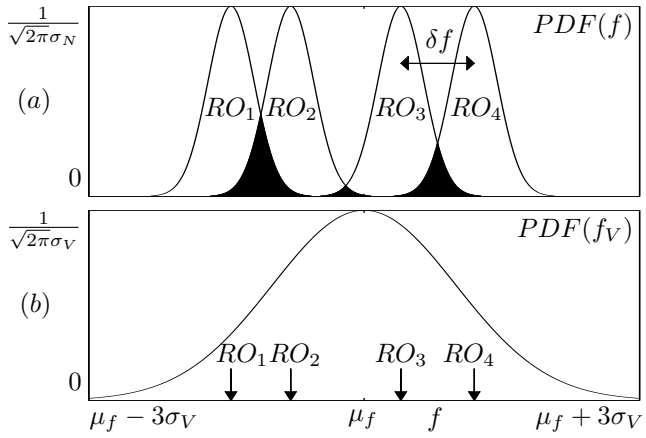

Fig. 13. RO PUF repeatability attack. (a) Reference ROs with overlapping frequency distributions, due to noise. (b) Normal distribution of mean RO frequencies, due to variance.

VII. Faster Modeling via Environmental Changes

So far, response repeatability imperfections originate from CMOS device noise. We increase the fraction of unstable responses U via environmental changes. So we switch from a side channel to a fault injection approach. We still perform multiple measurements per challenge $\vec{c}_i$, as required to compute a repeatability R, but we now use L different environments as depicted in figure 14. More generally, one could collect multiple response bits per environmental setting, but we limit ourselves to $M = 1$ for our repeatability attacks. For our preceding environmental study, we use larger values of M to suppress noise (majority vote). However, data of different environments is not 'mixed' then, but kept separated.

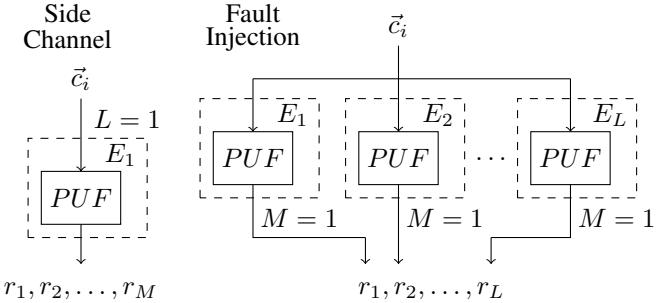

Fig. 14. Measurement setup for repeatability modeling attacks via environmental changes.

There are two basic data acquisition strategies for the attack. Either one handles the challenges one by one, continuously changing the environment. Or one applies all challenges before performing an environmental change. We opted for the latter strategy because of its convenience. There is no need to store all challenges. We employed an on-the-fly approach where

we re-seed a pseudorandom number generator with a constant value.

We are particularly interested in CRPs with $|\Delta x_V - \Delta x_B|$ slightly too large to pose repeatability issues due to CMOS device noise alone. Environmental changes modify delay, noise and bias characteristics of a PUF. The impact increases with the magnitude of the environmental deviation and is not necessarily balanced for all three parameters of interest. So additional CRPs are expected to become unstable when gathering repeatability data from different environments.

An example is given in figure 15 for $L = 3$. A challenge $\vec{c}_i$ evaluates to $r = 1$ under nominal conditions (E_1), in a repeatable manner. Variance Δx_V, noise Δx_N and bias Δx_B all decrease for environment E_3, with slightly different scaling factors for each. The same challenge still evaluates to $r = 1$, in even more stable manner. For E_2 however, where the impact is increasing, the challenge will regularly evaluate to $r = 0$.

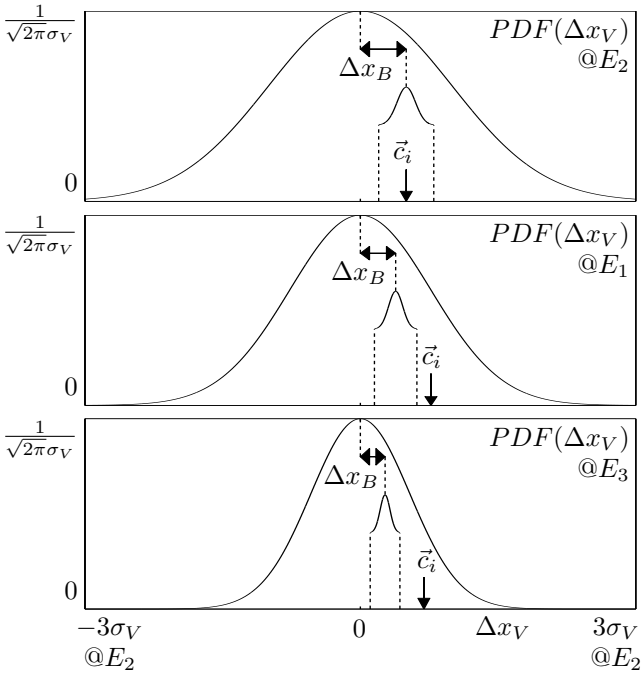

Fig. 15. Environmental changes increase the fraction of unstable CRPs.

We perform experimental measurements for all PUFs to support the former theory. There is no need to discuss RO and RO sum PUFs separately as their analog core is identical. But first we define the nominal PUF environment and we discuss the extent of deviation.

A. Environmental Experiments

Environmental deviations are chosen symmetrical around the nominal environment E_1, as in the end we want to obtain a model for a PUF under nominal conditions. We define the environment as the DC supply voltage V_S and the temperature T. Other environmental influences are not investigated here and are suggested as further work. We quantify experimentally the environmental impact on our PUFs, separately for the supply voltage and temperature case.

The nominal supply voltage of our 65nm chip equals 1.20V. We perform a sweep from 0.95V to 1.45V in steps of 0.05V, corresponding to $L = 11$ environmental settings. The magnitude of the deviations is limited by the operability of the chip. For the maximum voltage, we should also make sure not to damage any circuitry.

We define the nominal temperature as 20 °C. Similar as for supply voltage, a sweep is performed from −20 °C to 60 °C in steps of 10 °C, corresponding to $L = 9$ environmental settings. A TestEquity Half Cube temperature chamber is hereby employed. Again, operability and potential damage of the chip should be taken into account.

B. Arbiter PUF

For each environmental configuration, we query 32 arbiter PUFs with the same 2000 randomly chosen challenges, using $M = 15$. Figure 16 and 17 show the averaged supply voltage and temperature impact respectively. The fraction of usable CRPs U is rather independent of the environment, as shown in subfigures a. So all environments are to be considered as equally stable.

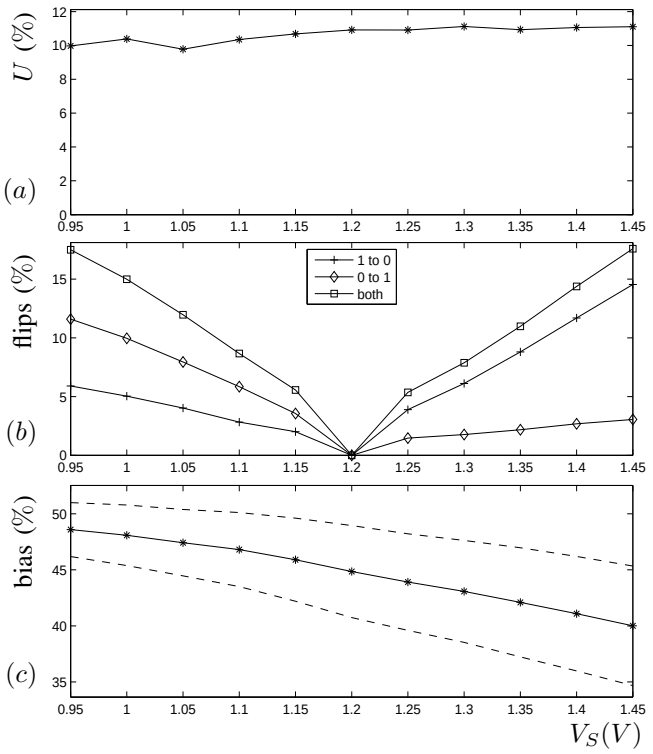

Fig. 16. Arbiter PUF: deviations around the nominal supply voltage $V_S = 1.20V$. (a) Fraction of unstable CRPs U. (b) Response bit flips with respect to the nominal environment. (c) Response bias.

However, we are interested in response instabilities across different environments, instead of within a single environment. For each PUF instance, each challenge and each voltage configuration, we first perform a majority vote to suppress CMOS device noise somewhat. Subsequently we determine the ratio of flipping response bits with respect to the nominal environment, as shown in subfigures b.

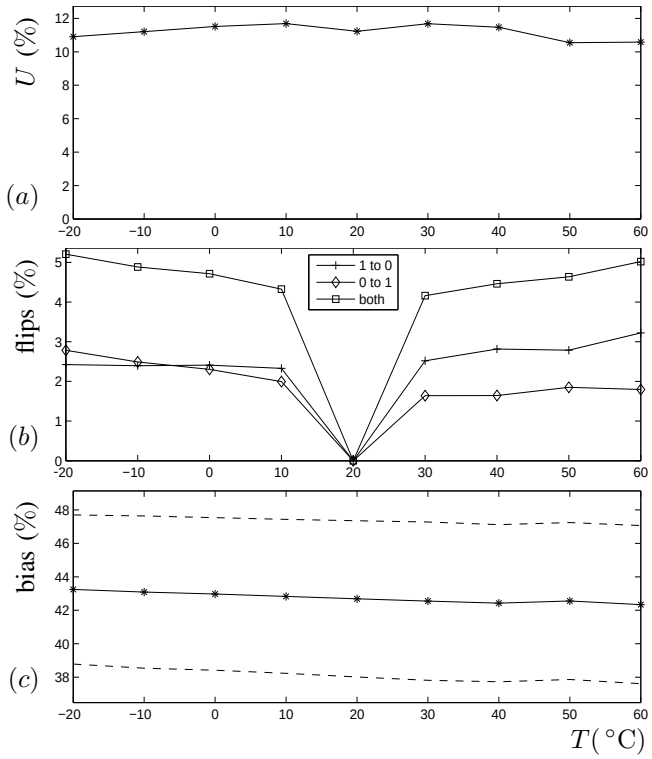

Fig. 17. Arbiter PUF: deviations around the nominal temperature $T = 20\,^{\circ}\mathrm{C}$. (a) Fraction of unstable CRPs U. (b) Response bit flips with respect to the nominal environment. (c) Response bias.

For the supply voltage case, we observe a significant amount of erroneous response bits when changing the environment. The further from the nominal environment, the more CRPs become unstable. Immediately around the nominal environment, there seems to be a larger increase in flipping bits. This is an artifact however, originating from CMOS device noise still present after the majority vote. For the temperature case, we observe only a very minor impact of the environment. So for our experimental fault injection attack later on, we only consider supply voltage deviations.

We also make a distinction between '1' to '0' and '0' to '1' flipping bits. For the supply voltage case, the bias increasingly shifts towards $r = 0$ with increasing supply voltage. This effect is shown more clearly in subfigure c, plotting the fraction of PUF responses evaluating to '1'. The dashed lines represent the ± 1 standard deviation interval with respect to the 32 PUF instances.

C. RO and RO Sum PUF

For each environmental configuration, we collect counter values for 2048 ring oscillators, using $M = 15$. Figure 18 and 19 show the averaged supply voltage and temperature impact respectively. As before, the fraction of usable CRPs U is rather independent of the environment, as shown in subfigures a.

We perform 2047 randomly chosen but independent frequency comparisons to evaluate the RO PUF behavior. The ratio of flipping response bits, with respect to the nominal environment, is shown in subfigures b. Our conclusions are similar as before: for the supply voltage case we observe a much larger impact than for the temperature case. So again, we only consider supply voltage deviations for our experimental modeling attacks later on.

Bias turns out to be limited and rather independent of the environment. As shown in subfigures c, bias is mainly introduced by comparing equal counter values: the PUF should return either a '1' or a '0' then. Therefore, we expect bias to decrease for the RO sum PUF, as the standard deviation of Δf_V increases with a factor $\sqrt{k}$ with respect to δf_V. Repeatability results are expected to be similar however as the noise standard deviation increases with a factor $\sqrt{k}$ as well.

Subfigures d show that the mean frequency is rather independent of the environment. The dashed lines represent the ± 1 standard deviation interval. However, for the supply voltage case, there is an either increasing or decreasing trend for the counter values of individual ring oscillators, causing response bits to flip. The environmental impact relative to the separate ring oscillator, determining oscillation time, is of a major importance hereby.

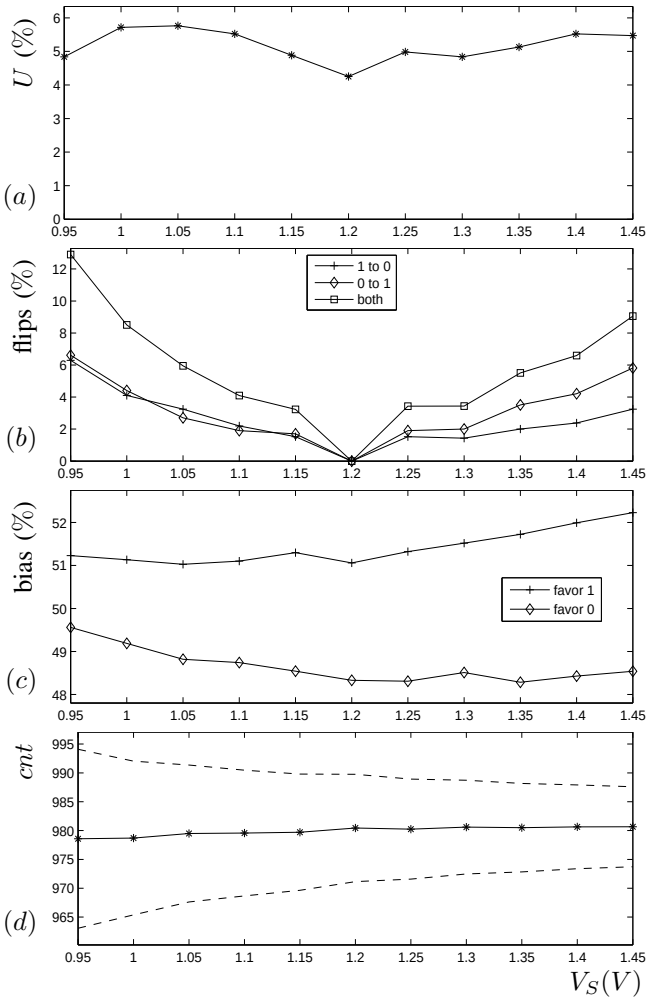

Fig. 18. RO (sum) PUFs: deviations around the nominal supply voltage $V_S = 1.20V$. (a) Fraction of unstable CRPs U. (b) Response bit flips with respect to the nominal environment. (c) Response bias. (d) Mean counter value cnt.

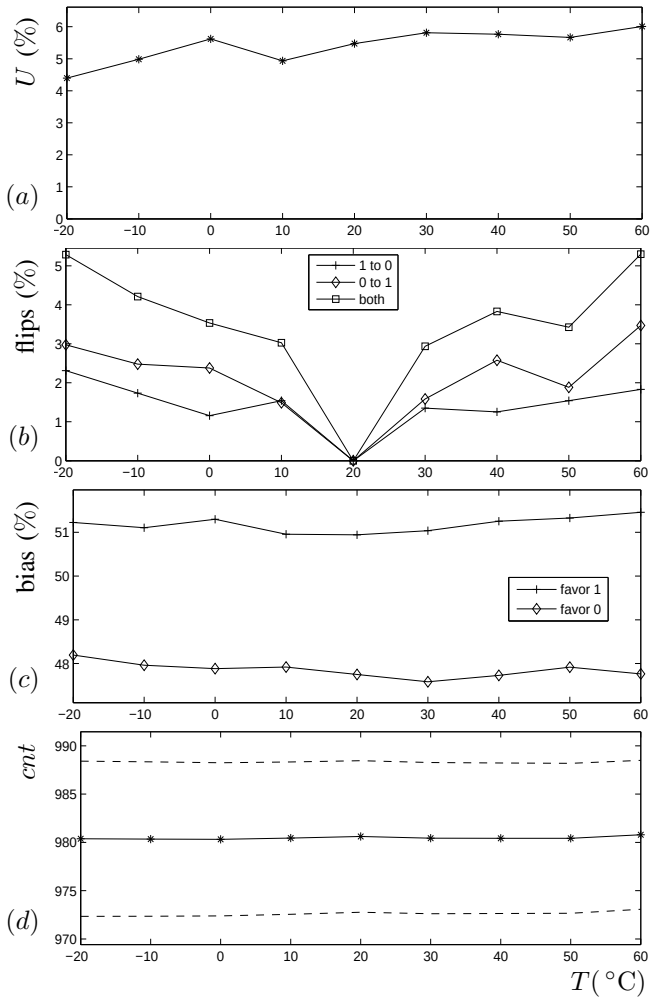

Fig. 19. RO (sum) PUFs: deviations around the nominal temperature $T = 20\,^{\circ}\mathrm{C}$. (a) Fraction of unstable CRPs U. (b) Response bit flips with respect to the nominal environment. (c) Response bias. (d) Mean counter value cnt.

VIII. Results and Discussion

We want an accurate model while keeping the number of PUF evaluations (LMN/U) low. We first determine an optimal value for parameter M, in case of the side channel approach ($L = 1$). For ease of comparison, we will employ the same value for L in case of the fault injection approach ($M = 1$). Subsequently we tabulate and compare accuracy results for both the arbiter and RO sum PUF. We conclude that environmental changes increase the fraction of useful CRPs U without deteriorating the modeling accuracy. As mentioned before, we only present supply voltage results.

A. Parameter M (side channel approach)

The larger M and N, the more accurate one can model. Figure 20 demonstrates that the best accuracy-performance trade-off is obtained for very low values of M. We plot the modeling accuracy versus the number of usable PUF evaluations (MN) for $M \in \{3, 15, 29\}$. Data is averaged over 32 arbiter PUF instances, but we expect similar behavior for RO sum PUFs. The accuracy verification set counts 5000 CRPs, improved in quality via majority voting ($M = 30$).

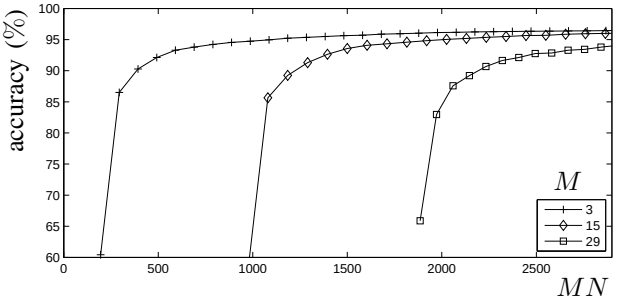

Fig. 20. Arbiter PUF modeling accuracy versus parameters M and N.

The first model can be constructed for $N = k+1 = 65$ and the modeling accuracy ramps up rapidly with increasing N afterwards. That's because repeatability measurement errors are dealt with efficiently. The ramp up behavior is conform with the condition number simulation in figure 12.

The fraction of usable CRPs U is approximately constant for large M. However, for high-performance attacks, we are particularly interested in low values of M: the dependency $U(M)$ is shown in figure 21. Data is averaged over 25000 CRPs and 32 arbiter PUF instances. Again, we expect similar behavior for RO sum PUFs.

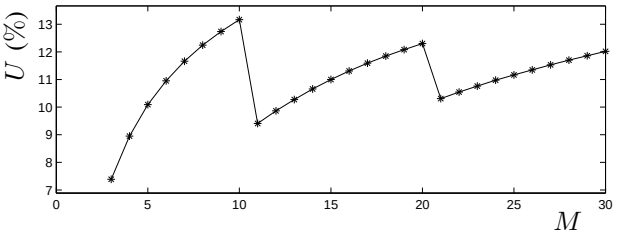

Fig. 21. Arbiter PUF: fraction of usable CRPs U as a function of M.

The dependency $U(M)$ is mainly caused by the asymmetry of discrete repeatability bins around $R = 10\%$ and $R = 90\%$. The larger M, the smaller the spacing between neighboring bins, the smaller the deviations around the constant value for very large M. Regarding overall performance, we expect the conclusions drawn from figure 20 to be dominant. Therefore we choose $M = 3$ for all further results, despite the worst-case U.

B. Accuracy and Performance

We measure modeling accuracy via a verification set of 3000 and 5000 randomly chosen CRPs for the arbiter and RO sum PUF respectively. Each verification challenge was evaluated M = 15 times on 32 arbiter PUF instances and one RO sum PUF instance respectively, with $M/2$ as a response bit decision threshold. Verification data is acquired for the nominal environment E_1 ($V_S = 1.20V$).

Table I and II provide accuracy-performance results for the arbiter and RO sum PUF respectively. For the arbiter PUF, accuracies are averaged over 32 PUF instances: the mean value and the ± 1 standard deviation interval is given. The leftmost

column of values corresponds with the side channel approach and serves as a reference.

Comparing the arbiter side-channel results with [5], we observe a slight decrease in accuracy. We indicate three possible root causes. First, another specimen of the same chip was employed for our measurements. Second, another sequence of random challenges was applied. Third, parameter M of the verification set has been decreased significantly. We do not consider this observation as very important however: our main interest is the accuracy-performance comparison with the fault-injection method.

	side channel	fault injection				
M	3	1				
L	1	3				
E	$1.20V$	$1.15V$ $1.20V$ $1.25V$	$1.10V$ $1.20V$ $1.30V$	$1.05V$ $1.20V$ $1.35V$	$1.00V$ $1.20V$ $1.40V$	$0.95V$ $1.20V$ $1.45V$
N \ U	7.5%	8.4%	10.3%	12.9%	15.4%	17.9%
100	85.1% ±9.9%	87.4% ±6.9%	88.7% ±3.8%	89.3% ±2.6%	88.3% ±2.5%	87.4% ±3.2%
200	93.6% ±1.5%	93.8% ±1.4%	93.9% ±1.2%	93.7% ±1.0%	93.6% ±1.0%	93.4% ±1.3%
300	94.6% ±1.2%	94.7% ±0.9%	95.0% ±0.7%	95.0% ±0.7%	95.0% ±0.7%	94.8% ±0.7%
400	95.3% ±0.8%	95.3% ±0.7%	95.6% ±0.7%	95.6% ±0.6%	95.6% ±0.5%	95.3% ±0.5%
500	95.6% ±0.6%	95.7% ±0.6%	95.9% ±0.6%	95.7% ±0.6%	95.9% ±0.5%	95.7% ±0.5%

TABLE I
ARBITER PUF: MODELING ACCURACY AND PERFORMANCE, WITH AND WITHOUT ENVIRONMENTAL DEVIATIONS.

	side channel	fault injection				
M	3	1				
L	1	3				
E	$1.20V$	$1.15V$ $1.20V$ $1.25V$	$1.10V$ $1.20V$ $1.30V$	$1.05V$ $1.20V$ $1.35V$	$1.00V$ $1.20V$ $1.40V$	$0.95V$ $1.20V$ $1.45V$
N \ U	3.7%	4.4%	5.7%	6.9%	7.5%	8.7%
100	96.1%	94.5%	95.2%	96.1%	95.4%	92.4%
200	98.0%	98.3%	97.7%	97.6%	97.8%	96.4%
300	98.4%	98.6%	98.4%	98.5%	98.2%	97.4%
400	98.8%	99.0%	98.6%	98.7%	98.6%	98.0%
500	98.8%	98.9%	98.8%	99.0%	98.9%	98.2%

TABLE II
RO SUM PUF: MODELING ACCURACY AND PERFORMANCE, WITH AND WITHOUT ENVIRONMENTAL DEVIATIONS.

The more to the right in both tables, the larger the supply voltage deviation. As discussed before, environmental deviations are chosen symmetrical around the nominal value. We observe no significant impact on the accuracy, with respect to the side channel case. This is especially clear for table I, which values are averaged. For the largest environmental deviation, the modeling speed improves with an equal factor $17.9/7.5 \approx 8.7/3.7 \approx 2.4$ for both arbiter and RO sum PUFs, while maintaining an excellent accuracy.

ML attacks are still faster, as demonstrated for arbiter PUFs using the same 65nm chip [4], although the gap is closing. We propose joined efforts instead of competition however: the presented techniques might facilitate a ML attack. Repeatability R is a much more informative PUF output than the (noisy) response bit r, so learning capabilities can only benefit. Further research is therefore recommended. Other fault injection techniques, applied either individually or simultaneously, might increase fraction U and hence the attack speed as well.

IX. FURTHER WORK

We are convinced that more research is required to fully understand the attack capabilities. Other strong PUF designs (e.g. the many arbiter PUF variants) should be examined for potential weaknesses. We also proposed joined efforts with ML techniques: repeatability data might be employed to facilitate automatic learning.

Alternative techniques for fault injection could be investigated as well. Consider for instance an unstable power supply, where more noise is literally injected into the analog PUF circuit. Electromagnetic radiation might be another idea worth exploring. Also remark that one could use multiple fault injection techniques simultaneously.

Countermeasures are another line of research. Additional hardware for on-chip error correction, as in a controlled PUF, might help. Also, one could try to detect different forms of fault injection. One can ensure that the PUF is only operational if the provided supply voltage is very close to the nominal value, for instance.

X. CONCLUSION

Repeatability measurements provide a new strategy for attacking strong PUFs in a CRP-based authentication application. Either as a pure side channel technique or possibly accelerated via fault injection. By applying environmental deviations, supply voltage in particular, response bit faults were introduced. As such we were able to increase the fraction of unstable CRPs, hereby improving the attack efficiency. As experimentally quantified using a 65nm CMOS chip, modeling attacks are accelerated with a factor 2.4 for both arbiter and RO sum PUFs, while maintaining an excellent accuracy.

ACKNOWLEDGMENT

This work was supported in part by the European Commission through the ICT programme under contract FP7-ICT-2011-317930 HINT. In addition this work is supported by the Research Council of KU Leuven: GOA TENSE (GOA/11/007), by the Flemish Government through FWO G.0550.12N and the Hercules Foundation AKUL/11/19. Jeroen Delvaux is funded by IWT-Flanders grant no. 121552.

REFERENCES

[1] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith, "Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data," *SIAM J. Comput.*, vol. 38, no. 1, pp. 97-139, Mar. 2008.

[2] B. Gassend, M. van Dijk, D. E. Clarke, E. Torlak, S. Devadas and P. Tuyls, "Controlled physical random functions and applications," *ACM Trans. Inf. Syst. Secur.*, vol. 10, no. 4, pp. 1-22, Jan. 2008.

[3] C. Helfmeier, C. Boit, D. Nedospasov and J.-P Seifert, "Cloning Physically Unclonable Functions," in *IEEE Int. Symposium on Hardware-Oriented Security and Trust*, HOST 2013, pp. 1-6, Jun. 2013.

[4] G. Hospodar, R. Maes, and I. Verbauwhede, "Machine Learning Attacks on 65nm Arbiter PUFs: Accurate Modeling poses strict Bounds on Usability," in *IEEE International Workshop on Information Forensics and Security*, WIFS 2012, pp. 37-42, Dec. 2012.

[5] J. Delvaux and I. Verbauwhede, "Side Channel Modeling Attacks on 65nm Arbiter PUFs Exploiting CMOS Device Noise," in *IEEE Int. Symposium on Hardware-Oriented Security and Trust*, HOST 2013, pp. 137-142, Jun. 2013.

[6] P. Koeberl, R. Maes, V. Rožić, V. Van der Leest, E. Van der Sluis, and I. Verbauwhede, "Experimental evaluation of Physically Unclonable Functions in 65 nm CMOS," in *IEEE European Solid-State Circuit conference*, ESSCIRC 2012, pp. 486-489, Sep. 2012.

[7] A. Konczakowska and B. M. Wilamowski, "Noise in Semiconductor Devices," Industrial Electronics Handbook, vol. 1 Fundamentals of Industrial Electronics, 2nd Edition, chapter 11, CRC Press 2011.

[8] K. Kuhn, C. Kenyon, A. Kornfeld, M. Liu, A. Maheshwari, W. Shih, S. Sivakumar, G. Taylor, P. VanDerVoorn and K. Zawadzki, "Managing Process Variation in Intel's 45nm CMOS Technology," *Intel Technology Journal*, vol. 12, no. 2, pp. 92-110, Jun. 2008.

[9] J.W. Lee, D. Lim, B. Gassend, G.E. Suh, M. van Dijk, and S. Devadas, "A technique to build a secret key in integrated circuits for identification and authentication applications," in *IEEE Symposium on VLSI Circuits*, VLSIC 2004, pp. 176-179, Jun. 2004.

[10] D. Merli, D. Schuster, F. Stumpf and G. Sigl, "Semi-invasive EM attack on FPGA RO PUFs and countermeasures," in *Workshop on Embedded Systems Security*, WESS 2011, pp 1-9, Oct. 2011.

[11] D. Merli, D. Schuster, F. Stumpf and G. Sigl, Side-channel analysis of PUFs and fuzzy extractors in *Trust and Trustworthy Computing conference*, TRUST 2011, pp. 33-47, Jun. 2011.

[12] U. Rührmair, F. Sehnke, J. Sölter, G. Dror, S. Devadas and J. Schmidhuber, "Modeling attacks on physical unclonable functions," in *ACM conference on Computer and Communications Security*, CCS 2010, pp. 237-249, Oct. 2010.

[13] U. Rührmair, S. Devadas and F. Koushanfar, "Security based on Physical Unclonability and Disorder," Introduction to Hardware Security and Trust, Springer, Book Chapter, 2011.

[14] S. Skorobogatov, "Semi-invasive attacks - a new approach to hardware security analysis, Technical Report UCAM-CL-TR-630, University of Cambridge, Computer Laboratory, Apr. 2005.

[15] G.E. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," in *IEEE Design Automation Conference*, DAC 2007, pp. 9-14, Jun. 2007.

[16] D. Suzuki and K. Shimizu, "The Glitch PUF: A New Delay-PUF Architecture Exploiting Glitch Shapes," in *Cryptographic Hardware and Embedded Systems*, CHES 2010, pp. 366-382, Aug. 2010.

[17] P. Tuyls, G.J. Schrijen, B. Skoric, J.V. Geloven, N. Verhaegh and R. Wolters, "Read-Proof Hardware from Protective Coatings," in *Cryptographic Hardware and Embedded Systems*, CHES 2006, pp. 369-383, Oct. 2006.

[18] M.-D. Yu and S. Devadas, "Recombination of physical unclonable functions, in *Government Microcircuit Applications & Critical Technology Conference*, GOMACTech 2010, Mar. 2010.

Jeroen Delvaux is pursuing the Ph.D. degree from the COSIC research group of the Electrical Engineering department of the KU Leuven in Belgium. Before, he worked 19 months as an R&D engineer for KLA-Tencor. He received the Master degree in EE from the KU Leuven in 2010. He also obtained a postgraduate in biomedical engineering from the KU Leuven in 2012. His main research interests include physically unclonable functions and efficient implementations of cryptographic hardware.

Ingrid Verbauwhede is a professor in the research group COSIC of the Electrical Engineering department of the KU Leuven in Belgium. At COSIC, she leads the embedded systems and hardware group. She is also associate professor at the EE department at UCLA, Los Angeles, CA. She joined COSIC in 2003 and UCLA in 1998. Before, she was a post-doctoral visiting researcher and lecturer at UC Berkeley and she worked as a principal engineer for TCSI and Atmel in Berkeley, CA. She received her PhD degree from KU Leuven and IMEC in 1991. She is a Member of IACR and she was elected as member of the Royal Flemish Academy of Belgium for Science and the Arts in 2011. Her main interest is in the design and the design methods for secure embedded circuits and systems. She is a fellow of IEEE.